DAWGPOOL

Group Number: 6

Sammy Bharadwaj, Liuhan Ke, Tony Ngo, Keaton Staggs, Kevin Tat

Figma Link: DAWGPOOL Figma Prototype

Will you create an interactive prototype before you begin coding (e.g. using Figma), or will you work from static wireframes?

We will work from static wireframes since the pages we have now are clear-cut and don't have to be interactive so that the users can understand the flow of the web app.

What are the acceptance tests that your team will perform before beginning user testing? Describe the acceptance testing process for at least two key features, including which results would indicate a successful test and which results would indicate a failed test.

Test Case 1: Verifying Information Loaded on the Main Page

- **Objective**: Ensure a new user's information can be successfully loaded and displayed on the main page after registration.
- **Process**:
    1. A new user logs into the website (user authentication should be functional at this stage).
    2. The user enters their personal information, which should appear on the main page, allowing other users to find potential carpool partners.
    3. We verify whether the entered information is accurately rendered on the main page.
- **Expected Outcome**:
    1. **Pass**: The user's information appears correctly on the main page, visible to other users.
    2. **Fail**: The user's information fails to load or is rendered incorrectly on the main page.

Test Case 2: Testing the Search Filter Functionality

- **Objective**: Confirm that the search filter displays only relevant users based on the selected region, enabling users to locate nearby carpool options.
- **Process**:
    1. A JSON file containing dummy data is created for testing purposes.
    2. Using the search box, we test whether selecting a specific region updates the main page to display only users from that region.
- **Expected Outcome**:
    1. **Pass**: The main page dynamically updates to display only users who match the selected region.
    2. **Fail**: The main page remains static, displaying all users regardless of the search filter selection.

Test Case 3: Testing the Real-time Messaging Functionality

- **Objective**: Confirm that the messaging feature allows users to communicate with each other in real time.
- **Process**:
    1. One user logs in through email authentication, creates an account and messages another authenticated user through the platform.
    2. We verify through the messaging page that user messages are being received and displayed correctly on the chat view.
- **Expected Outcome**:
    1. **Pass**: The messaging page dynamically updates to display user messages as they are sent.
    2. **Fail**: Users do not receive messages from other users in the chat view of the messaging page.

What are the limitations of your acceptance tests? List some of the ways that your team's in-house testing environment may differ from the context in which your expected users will be interacting with the product.

- When doing the first testing to see if the website can load new data properly, users might not input their data immediately but first explore the website;
- When users are interacting with the website, they might search for something else other than region so that the searching function could fail.

How will you conduct user testing?

Our ideal demographic for the user testing would be students who commute to school so we plan on sending out a survey for students to fill out and reimburse them for their time. We will then conduct user testing by interviewing students at UW and having them perform a set of tasks on our app. Through this trial, we can observe how they progress through the user flow and see if it is efficient or not. We can also take note of any pain points that they may endure during the testing process. Once we have 5-10 tests, we'll take the data and make necessary changes to ensure the user experience is solid.

How will you decide which bugs to fix first?

We will decide which bugs to fix first by understanding their **severity**. To measure severity, we will consider how each bug impacts the usability of our application. To categorize each bug, we will place them into categories of **high, medium, and low severity**. High-severity bugs will be issues that interfere with the usability of our primary functionality, such as our matching system. If a bug occurs that prevents our matching functionality from working properly, users will be unable to connect with others and complete their goal of ridesharing together. Medium-severity bugs are categorized as issues affecting features that fall outside of our must-have functionality. For instance, if a user is messaged through the platform by a potential match and fails to receive an email notification. Since the main functionality of the application is not affected, this bug would be medium severity. Last, we will categorize low-severity bugs as minor issues that have no impact on users achieving their goals. For instance, if we have an image within the application that is slightly off-center. A misaligned image would not affect users completing the main functionality but should be addressed at some point.

How will you re-test the solution after the bug fixes have been completed?

After each bug fix, our team will rerun the relevant acceptance tests to confirm that the bug is fixed without new problems. Additionally, we will conduct user acceptance testing on top of the existing relevant acceptance test. We will try to find a small group of users to test our apps, by doing this we may be able to find edge cases and issues. Lastly, we will implement end-to-end tests to simulate real user interactions across the app. By implementing end-to-end testing, we will be able to ensure the solution is functioning properly after the bug fixes.